Evaluating Entropy for TRNGs: Fast, Robust and Provably Secure

Abstract

Estimating entropy for true random number generators is a task of critical importance, because feeding cryptographic applications with insufficient entropy leads to poor security. This is a challenging task as the entropy needs to be estimated at high accuracy, high confidence level and with possibly minimal number of samples (ideally no more than needed for extraction),

In this paper we analyze the performance of a simple collision-counting estimator, and show that it is much better than min-entropy estimators or Shannon-entropy estimators studied before in this context, and moreover that it is robust against changes in the source distribution (due environmental conditions or adversarial influences). More precisely, we show an estimator which for n samples, and confidence $1-\epsilon$:

(a) Extremely efficient: reads the stream in one-pass and uses constant memory (forward-only mode)
(b) Accurate: estimates the amount of extractable bits with a relative error $O(n^{-\frac{1}{2}}\log(1/\epsilon))$, when the source outputs are i.i.d.
(c) Robust: keeps the same error when the source outputs are independent but the distribution changes up to $t = O(n^{\frac{1}{2}})$ times during runtime

We demonstrate that the estimator is accurate enough to adjust post-processing components dynamically, estimating entropy on the fly instead investigating it off-line.

1 Introduction

1.1 Estimating Entropy for True Random Number Generators

True Random Number Generators are devices that utilize some underlying physical process to generate bits that are statistically indistinguishable[1] from uniform (called truly random). Examples of such processes are radio noise [Haa], radiation [Wal], thermal noise [JK99] or noise from sensors in mobile devices [BKMS09, BS]. A typical TRNG consists of an entropy source, harvesting mechanism and a post-processing[2] component which reduces bias and correlations present in raw data [Sun09, TBK$^+$]. In reality, outputs of available sources are somewhat unpredictable but biased, therefore the output quality depends critically on adjusting the post-processing part to the source. Basically, post-processing functions give provable guarantees on the output quality when fed with inputs of sufficiently high entropy (well known examples are universal hash functions [BST03] or the von Neumann extractor [vN51]). Therefore, to achieve the required quality we need to estimate the entropy in the source. This requirement is not only a matter of provable security, but a serious practical concern as low entropy may lead to attacks on real-world applications [dRHG$^+$99]. Recent examples are bugs in the Linux Random Number Generator on Debian distributions [GPR06] on Android distributions [KKHD14]. For this reason, entropy evaluation is considered a necessary part of the developing designing process and is strongly recommended by standards [TBK$^+$].

[1] Which means closeness in the variational distance (distance ϵ smaller than 2^{-80} for practical applications)
[2] Sometimes called also the conditioning component [TBK$^+$], or an extractor in the theoretical literature.

To formulate the problem more precisely, we assume without losing generality that the entropy source output is already digitized and forms a sequence of symbols $X_1, \ldots, X_n$ over a finite alphabet $\mathcal{X}$ (for example $\mathcal{X}$ may be the set of 10-bit strings), which are produced by repeating the generating procedure in consecutive time intervals. Keeping in mind that there are many entropy notions and not all of them are suitable for cryptographic purposes (such as generating random numbers) we can state the following problem

> **Problem:** How to estimate the (cryptographically relevant) entropy of a stochastic process $X_1, X_2, \ldots, X_n$ over an alphabet $\mathcal{X}$?

The right entropy notion here is not the popular Shannon entropy, but the more conservative entropy notion called *min-entropy*. The theory of randomness extraction characterizes min-entropy as the measure of the amount of almost uniform bits that can be extracted (using best post-processing functions) from a given distribution [RW05, RW05]. More specifically, the min-entropy is the negative logarithm of the likelihood of the most heavy element: if every outcome appears with probability at most 2^{-k} then we say that the distribution has k-bits of min-entropy.

Estimating the min-entropy of a TRNG source is also recommended by standards [TBK$^+$]. Once we know how much entropy we have collected, we can tune the parameters of an appropriate extractor (post-processing) function and produce an almost biased string. The task of estimating entropy is independent on a specific post-processing function. Different post-processing functions yield only different trade-offs between the input entropy and the output length and quality (also in memory and time resources consumed). However, estimating entropy is a non-trivial problem because one needs to find a right balance between the accuracy, sample complexity and the source model.

1.2 Related Works

Standards like the most recent NIST recommendation [TBK$^+$] suggest to approximate the min-entropy based on empirical frequencies plugged into the entropy formula and many independent samples. This approach is applied in works with focus on provable security [BST03, BKMS09, VSH11]. We note that this process requires a large amount of extra memory, namely one needs to compute the frequencies of all elements $x \in \mathcal{X}$. This costs $\Omega(|\mathcal{X}|)$ bits of memory, actually more if we want an entropy estimate with a small relative error δ. For this we need to keep the frequencies with a precision up to $\frac{\delta}{|\mathcal{X}|}$, which means $\Omega(|\mathcal{X}| \log\left(\frac{|\mathcal{X}|}{\delta}\right))$ bits. For 30-bit blocks this is more than 4GB of memory.

In general, the "plug-in" estimator is not memory-efficient on small mobile or embedded devices [LRSV12]. The authors of the referenced work proposed to construct an estimator for Shannon entropy instead of min-entropy, which basically just quickly reads the stream and operates within a constant amount of memory. However, this is not provably secure unless for stateless sources (producing i.i.d. symbols) as shown by a result called the Asymptotic Equipartition Property [Sha48, Ash90]. The price is a wide error margin for which the best known bound is only $O(|\mathcal{X}|)$ [Hol06], which is more than 1000 bits for a source with only 10-bit outputs blocks.

1.3 Our Contributions

We present a simple estimator based on a new idea of *collisions counting* which operates in *constant memory*. Technically speaking, we estimate not the min-entropy but a slightly weaker notion called collision entropy, which turns out to be close enough. Using the so called *entropy smoothing* we can go back from collision entropy to min-entropy losing only $\log(1/\epsilon)$ bits where ϵ is the chosen security parameter, typically $\epsilon = 2^{-80}$. Moreover, for most popular post-processing functions based on universal hashing [BST03, VSH11] our estimate can be applied with no loss as if it was min-entropy, because universal hash functions work with collision entropy [HILL99]. The pseudocode is given in Algorithm 1.

Algorithm 1: Collision Entropy Estimator

```
Data: i.i.d. samples $x_1, \ldots, x_n$ from $X \in \{0,1\}^m$
Result: An estimation of $\mathbf{H}_2(X)$
1 $P \leftarrow 0$
2 for $i = 2, \ldots, n$ do
3     if $x_i = x_{i-1}$ then
4         $P \leftarrow P + 1$
5 $\hat{H} \leftarrow -\log_2\left(\frac{P}{n-1}\right)$
6 return $\hat{H}$
```

For this estimator, we prove the following key features:

(a) convergence bounds: we give clear error bounds on the estimator convergence, depending on the chosen security level (output indsitinguishability) and the number of samples. Namely, for n samples at confidence $1-\epsilon$ we estimate the entropy per bit with a relative error $\delta = O\left(\sqrt{\frac{|\mathcal{X}|\log(1/\epsilon)}{n}}\right)$, for $n = \Omega(|\mathcal{X}|\log(1/\epsilon))$. Moreover, the alphabet size $|\mathcal{X}|$ can be replaced by 2^{H_2} where H_2 is the collision entropy rate (collision entropy per block) of the source. For more details, see Theorem 1.

(b) provable security: using the estimate together with universal hash functions we extract all the entropy but $O\left(\sqrt{n2^{H_2}\delta}\right)$ bits (the result being at most $O(\epsilon)$-far from the uniform distribution). For more details, see Corollary 1.

(c) efficiency: by definition, the estimator works in one pass and constant memory, being extremely efficient for long streams of data. This way we improve previous heuristic on on-line entropy estimation [LPR11] with an estimator even more efficient and, in addition, provably secure.

(d) robustness in changing environments: we prove the convergence relaxing the i.i.d assumption. Namely, we require consecutive outputs to be independent but allow the source to "switch" its internal state t times, changing the output distribution ($t \ll n$). This result has two consequences: first, the estimator is robust against *environmental changes* (accidental or adversarial); second, it allows for checking entropy in *production environments* (online) where distributions may be different than estimated in laboratories. The importance of these features were discussed in CHES papers [BST03] and [BL05]. As for further applications, in Section 5 we show how to adapt our technique to estimate entropy of a source consisting of a few independent sources.

1.4 Source Model

The source must have a certain structure to allow for estimating entropy from samples with high accuracy and confidence (because we are interested in provable security) but on the other hand the model should be possibly general to cover a possibly large range of real-world applications. From a theoretical perspective, the most general approach is to model entropy sources by Markov chains of finite order [Mau92, TBK+]. This is, however, extremely

Estimator	Source Model	Robustness	Adaptive	Prov. Sec.
[BST03]	small family of sources	entropy-preserving switching	No	Yes
[BL05]	independent binary	changing bias	Yes	Yes
[LRSV12]	i.i.d symbols	fixed distribution	Yes	No
this paper	independent symbols	on-line distribution changes	Yes	Yes

Table 1 Our estimator compared to related works.

inefficient in terms of the complexity/accuracy trade-off [TBK$^+$]. In this work we adopt the common modeling approach, which assumes that the source outputs are i.i.d. [LRSV12, VSH11, BL05, BKMS09]. Our model is substantially stronger: we assume that the source distribution changes at most t times at arbitrarily chosen moments. That is

- $X_1, X_2, \ldots, X_n$ are independent
- The number of indexes $= 1, \ldots, ni$ such that $X_i \overset{d}{\neq} X_{i+1}$ is at most t

This model captures scenarios where environmental conditions change and influence the source during entropy harvesting.

1.5 Convergence Bounds

In the literature, estimating entropy is typically not given a rigorous treatment. The newest NIST recommendation [TBK$^+$] suggest to take $n \gg 10^6$ for empirical evaluations, which should be big enough to ensure convergences. Other works [VSH11] also evaluate entropy over huge data sets, like “overnight” samples. On the other hand our bounds show that, at least under our (relaxed) i.i.d. assumption, this number can be much smaller. In fact, we are able to estimate entropy in production environments, when we don’t have much more data except what is roughly necessary for extraction. This way we can make the statistical error small enough to not to affect provable security level we want to achieve (e.g. $\epsilon = 2^{-80}$).

1.6 Efficiency and Provable Security

The work [LPR11] discusses an on-line entropy estimation technique for TRNGs by approximating the source Shannon Entropy, under the i.i.d. assumption. This is however not secure, as in this setting Shannon Entropy can be converted to min-entropy only with a huge entropy loss $O(|\mathcal{X}|\sqrt{n}\log(1/\epsilon))$ [Hol06]. Also, no convergence result for the entropy estimator itself was given. Comparing to this work, we lose at most $\log(1/\epsilon)$ bits when converting to min-entropy (see Corollary 1) and provide a clear convergence bound with a loss at most $O(\sqrt{|\mathcal{X}|n\log(1/\epsilon)})$ for confidence $1-\epsilon$. Also, we use less memory as [LPR11] need a sliding window with a size that affects the estimator convergence.

1.7 Robustness in Changing Environments

In real world, devices that generate random numbers operate under varying environmental conditions and can be indirectly affected by a number of processes. The output distribution in the production environment may be different than during the testing stage. Examples may be temperature or voltage changes [BST03], or even different way the user is interacting with the device - for example, the quality of accelerometer-based TRNG depends on the “shaking pattern” [VSH11]. The issue becomes even more serious, where environmental parameters can be manipulated by a malicious adversary [BST03]. There are two ways to handle this issue: (a) trying to investigate all relevant factors during off-line tests, an provide a lower bound [VSH11] or (b) developing a design robust against changes in the entropy rate [BST03, BL05, LRSV12]. The second way seems to be more promising (and also more challenging) as addressing all factors that can influence the source is binded to a particular hardware and thus not a generic approach. Moreover, the approach (a) is a passive way of solving the issue whereas (b) can be used to actively monitor the device behavior in production environments. More concretely, detecting a decrease in the entropy rate may be a trigger raising an attack alarm [BL05]. Lastly, we may want to use the robustness to handle multiple sources which contribute synchronously but independently to outputs blocks (for example, readings from all accelerometer axes). A short example is discussed in Section 5.

1.8 Out techniques

Our approach is based on using large deviation inequalities and some Jensen inequalities.

1.9 Organization

In Section 2 we provide necessary definitions and useful inequalities. The convergence of the entropy estimator is discussed in Section 3. Some further applications are explained in Section 4 and Section 5.

2 Preliminaries

2.1 Information-theoretic divergence measures

▶ **Definition 1** (Variational (Statistical) Distance)**.** We say that discrete random variables X_1 and X_2, taking values in the same space, have the *statistical distance at least* ϵ if their probability mass functions are at most ϵ-away in terms of the ℓ_1 norm, that is

$$\sum_x |\mathbf{P}_{X_1}(x) - \mathbf{P}_{X_2}(x)| \leqslant \epsilon.$$

2.2 Entropy Notions

▶ **Definition 2** (Min-Entropy)**.** The min-entropy of a random variable X is defined as $\mathbf{H}_\infty(X) = \max_x \log \frac{1}{\mathbf{P}_X(x)}$.

▶ **Definition 3** (Collision-Entropy)**.** The collision-entropy of a random variable X is defined as $\mathbf{H}_2(X) = -\log\left(\sum_x (\mathbf{P}_X(x))^2\right)$.

▶ **Definition 4** (Shannon-Entropy)**.** The Shannon-entropy of a random variable X is defined as $\mathbf{H}(X) = -\sum_x \mathbf{P}_X(x) \log \mathbf{P}_X(x)$.

▶ **Definition 5** (Smooth Min-Entropy)**.** We say that X has k-bits of ϵ-smooth min-entropy if X is ϵ-close to Y such that $\mathbf{H}_\infty(Y) \geqslant k$.

▶ **Lemma 1** (From collision to smooth min-entropy [Cac97])**.** *Suppose that* $\mathbf{H}_2(X) \geqslant k$. *Then* $\mathbf{H}^\epsilon_\infty(X) \geqslant k - \log(1/\epsilon)$.

2.3 Randomness Extractors

Extractors are functions which process weak sources into distributions that are close (in the information-theoretic sense) to the uniform distribution. In general, they need some amount of auxiliary randomness called *seed*. The seed is passed as an extra argument in the definition.

▶ **Definition 6** (Seeded extractors). A deterministic function $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^k$ is a (k,ϵ)-extractor for X if we have

$$\mathrm{SD}\left(\mathrm{Ext}(X, U_d), U_d; U_k, U_d\right) \leqslant \epsilon$$

▶ Remark (Relaxing min-entropy for seeded extractors). The min-entropy required in the source can be relaxed at least in two ways:

(a) X needs to be only close to a distribution with entropy k
(b) The entropy notion can be collision entropy, instead of much more restrictive min-entropy.

▶ **Lemma 2** (A necessary conditions for extracting). *Suppose that* Ext *on input X outputs a distribution which is ϵ-close Let* Ext *be any function such that* $\mathrm{SD}\left(\mathrm{Ext}(X,S); U_k|S\right) \leqslant \epsilon$. *Then X is ϵ-close to a distribution of min-entropy at least k.*

▶ **Definition 7.** A family $\mathcal{H}$ of functions from n to m is called *universal*, if for a random member $H \in \mathcal{H}$ and every different $x, y \in \{0,1\}^n$ we have

$$\Pr[H(x) = H(y)] = 2^{-m}.$$

▶ **Lemma 3** (Universal families are good extractors). *Suppose that* $\mathbf{H}_2(X) \geqslant k + 2\log(1/\epsilon)$. *Let $\mathcal{H}$ be a universal family of functions from n to m bits. For any $x \in \{0,1\}^n$ and $h \in \mathcal{H}$ define*

$$\mathrm{Ext}(x,h) = h(x)$$

Then we have

$$\mathrm{SD}\left(\mathrm{Ext}(X,H), S; U_k, S\right) \leqslant \epsilon$$

where H is a random element of $\mathcal{H}$.

2.4 Useful inequalities

▶ **Lemma 4** (Jensen's Inequality). *Let I be an interval and $f : I \to \mathbb{R}$ be a convex function. Then we have*

$$\sum_{i=1}^{n} \alpha_i f(u_i) \geqslant f\left(\sum_{i=1}^{n} \alpha_i u_i\right)$$

for any numbers $u_1, \ldots, u_n \in I$ and non-negative weights $\alpha_1, \ldots, \alpha_n$ that sum up to 1.

▶ **Lemma 5** (Multiplicative Chernoff Bound []). *Let $X_1, \ldots, X_n$ be independent binary random variables. Define $\hat{p} = \frac{\sum_{i=1}^{n} X_i}{n}$. Then we have*

$$\Pr\left[\hat{p} \leqslant (1-\delta)\mathbf{E}\hat{p}\right] \leqslant \exp\left(-np\delta^2/2\right)$$

for any positive number δ.

3 Main Result

▶ **Theorem 1** (The estimator convergence). *Let $X_1,\ldots,X_n$ be independent random variables over a finite domain $\mathcal{X}$. Suppose that the distribution of X_i changes at most t times when i goes from 1 to n. Then for any ϵ the output $\hat{H}$ of Algorithm 1 run over $X_1,\ldots,X_n$ and the collision entropy of $H_2 = \frac{1}{n}\mathbf{H}_2(X_1,\ldots,X_n)$ satisfy*

$$\mathbf{H}_2(X_1,\ldots,X_n) \geqslant (\hat{H}-\delta)\cdot n.$$

with the relative error δ smaller than any of the two bounds below

$$\delta = \sqrt{\frac{2^{\hat{H}}\cdot 4\log(2/\epsilon)}{n}} + \frac{2^{\hat{H}}\cdot 4\log(2/\epsilon)}{n} + \frac{2^{\hat{H}}\cdot(t+1)}{n\ln 2} \tag{1}$$

$$\delta = \sqrt{\frac{2^{H_2}\cdot 4\log(2/\epsilon)}{n}} + \frac{2^{H_2}\cdot(t+1)}{n\ln 2}. \tag{2}$$

▶ **Corollary 1** (Provable security with any min-entropy extractor). *Let Ext be any (k,ϵ) extractor from X^n to $\{0,1\}^\ell$. Then the output of Ext on $X_1,\ldots,X_n$ is $O(\epsilon)$-close to the uniform distribution on $\{0,1\}^\ell$, provided that*

$$k \geqslant n\hat{H} - (n\delta + \log(1/\epsilon)).$$

were $\hat{H}$ and δ are as in Theorem 1. Moreover, for universal hash functions it's enough to assume

$$k \geqslant n\hat{H} - n\delta.$$

Proof. The proof follows immediately from the extractor definition and the conversion between smooth min-entropy and collision entropy (Lemma 1). For the extractor built on universal hash functions we simply use the fact that if they the assumption about k bits of min-entropy on input can be relaxed to k bits of collision entropy. ◀

▶ Remark. This corollary shows that our estimator can be coupled with any min-entropy extractor (post-processing function). The entropy loss is due to estimation statistical error plus up to $\log(1/\epsilon)$ bits for conversion (not needed for universal hash functions). Note that typically we have $n \gg \log(1/\epsilon)$ and therefore $\log(1/\epsilon) \ll n\delta$. Thus the entropy loss in Corollary 1 equals $O(\delta n)$

Proof of Theorem 1. Suppose that $X_1,\ldots,X_n$ are independent, and for some $t\in[0,T]$ there are numbers n_j satisfying

$$1 = n_0 < n_1 < n_2 \ldots < n_{t+1} = n+1 \tag{3}$$

such that for every $j = 0\ldots,t$ we have

$$\forall i \in \{n_j,\ldots,n_{j+1}-1\} \quad X_i \overset{d}{=} Y_j.$$

This corresponds to the scenario where the distribution of the source is switched at moments $n_1,\ldots,n_t$. Let $\hat{p}^j_{\mathrm{col}}$ be the collision probability estimate for samples X_i where $i = n_j,\ldots,n_{j+1}-1$. That is

$$\hat{p}^j_{\mathrm{col}} = \begin{cases} \frac{\sum_{n_j<i<n_{j+1}} \mathbf{1}_{\{X_i=X_{i-1}\}}}{n_{j+1}-n_j-1}, & n_{j+1}-n_j > 1 \\ 0, & n_{j+1}-n_j = 1 \end{cases} \tag{4}$$

Note that

$$\mathbf{E}\left[\hat{p}^{j}_{\text{col}}\right] = p^{j}_{\text{coll}} \tag{5}$$

where $p^{j}_{\text{coll}} = \text{CP}(Y_j)$. Let p_{col} be the collision probability estimate, computed by the algorithm, over samples $X_1, \ldots, X_n$. In other words

$$\hat{p}_{\text{col}} = \frac{\sum_{i=1}^{n-1} \mathbf{1}_{\{X_i = X_{i+1}\}}}{n-1} \tag{6}$$

Skipping in Equation (6) these indexes i for which $i = n_j - 1$ for some j, and using Equation (5) we obtain

$$\begin{aligned}
\mathbf{E}\left[\hat{p}_{\text{coll}}\right] &= \frac{1}{n-1} \sum_{i=n_j}^{n_{j+1}-1} \mathbf{E}\left[\mathbf{1}_{\{X_i = X_{i+1}\}}\right] \\
&> \frac{1}{n-1} \sum_{j=0}^{t} \sum_{i=n_j}^{n_{j+1}-2} \mathbf{E}\left[\mathbf{1}_{\{X_i = X_{i+1}\}}\right] \\
&= \frac{1}{n-1} \sum_{j=0}^{t} \left(n_{j+1} - n_j - 1\right) p^{j}_{\text{coll}}.
\end{aligned} \tag{7}$$

Note that thre random variables $Z_i = \mathbf{1}_{\{X_i = X_{i+1}\}}$ are not independent, so we cannot apply the Chernoff Bound directly. However, we can take advantage of the fact that the subsequences with odd and even indexes are independent. Define

$$\begin{aligned}
I_1 &= \{j :\ 1 \leqslant i \leqslant n-1,\ j \equiv 1 \mod 2\} \\
I_2 &= \{j :\ 1 \leqslant j \leqslant n-1,\ j \equiv 0 \mod 2\}
\end{aligned}$$

The estimate in Equation (6) can be expressed as a combination of estimates over the set I_1 and I_2, as follows

$$\hat{p}_{\text{col}} = \frac{|I_1|}{n-1} \cdot \hat{p}_1 + \frac{|I_2|}{n-1} \cdot \hat{p}_2 \tag{8}$$

where

$$\hat{p}_i = \frac{\sum_{i \in I_j} Z_i}{|I_j|}, \quad j = 1, 2.$$

By the Chernoff Bound applied separately to $\hat{p}_1$ and $\hat{p}_2$, we conclude that every of the following inequalities

$$\begin{aligned}
\mathbf{E}\left[\hat{p}_1\right] &\leqslant \hat{p}_1 + \sqrt{\frac{2\ln(2/\epsilon)\mathbf{E}\left[\hat{p}_1\right]}{|I_1|}} \\
\mathbf{E}\left[\hat{p}_2\right] &\leqslant \hat{p}_2 + \sqrt{\frac{2\ln(2/\epsilon)\mathbf{E}\left[\hat{p}_2\right]}{|I_1|}}
\end{aligned}$$

holds with probability $1 - \frac{\epsilon}{2}$. By the union bound, they are satisfied simultaneously with probability at least $1 - \epsilon$. Multiplying these inequalities by the weights $\frac{|I_j|}{n-1}$ for $j = 1, 2$

respectively, and using Equation (8) we obtain

$$\begin{aligned}
\mathbf{E}\hat{p}_{\mathrm{col}} &= \frac{|I_1|}{n-1}\cdot\mathbf{E}\left[\hat{p}_1\right] + \frac{|I_2|}{n-1}\cdot\mathbf{E}\left[\hat{p}_2\right] \\
&\leqslant \frac{|I_1|}{n-1}\cdot\hat{p}_1 + \frac{|I_2|}{n-1}\cdot\hat{p}_2 + \left(\frac{|I_1|}{n-1}\cdot\sqrt{\frac{2\ln(2/\epsilon)\mathbf{E}\left[\hat{p}_1\right]}{|I_1|}} + \frac{|I_2|}{n-1}\cdot\sqrt{\frac{2\ln(2/\epsilon)\mathbf{E}\left[\hat{p}_2\right]}{|I_2|}}\right) \\
&\leqslant \hat{p}_{\mathrm{coll}} + \frac{\left(\sqrt{2\ln(2/\epsilon)|I_1|\mathbf{E}[\hat{p}_1]} + \sqrt{2\ln(2/\epsilon)|I_2|\mathbf{E}[\hat{p}_2]}\right)}{n-1}. \qquad (9)
\end{aligned}$$

with probability $1-\epsilon$. In order to simplify the rest of the proof, we use the following convention: from now all the inequalities hold with probability $1-\epsilon$ unless stated otherwise. From Equation (9), by applying the inequality $\sqrt{a}+\sqrt{b}\leqslant\sqrt{2(a+b)}$ (which follows by the Jensen inequality) and Equation (8), we conclude that

$$\begin{aligned}
\mathbf{E}\hat{p}_{\mathrm{col}} &\leqslant \hat{p}_{\mathrm{coll}} + \frac{\sqrt{4\ln(2/\epsilon)(n-1)\mathbf{E}[\hat{p}_{\mathrm{coll}}]}}{n-1} \\
&= \hat{p}_{\mathrm{coll}} + \sqrt{\frac{4\ln(2/\epsilon)\mathbf{E}[\hat{p}_{\mathrm{coll}}]}{n-1}} \qquad (10)
\end{aligned}$$

To make the right-hand side independent of the unknown parameter $\mathbf{E}\hat{p}_{\mathrm{col}}$ we rewrite Equation (10) as

$$\left(\sqrt{\mathbf{E}\hat{p}_{\mathrm{col}}} - \sqrt{\frac{\ln(2/\epsilon)}{n-1}}\right)^2 \leqslant \hat{p}_{\mathrm{coll}} + \frac{\ln(2/\epsilon)}{n-1}$$

which implies

$$\begin{aligned}
\mathbf{E}\hat{p}_{\mathrm{col}} &\leqslant \left(\sqrt{\hat{p}_{\mathrm{coll}} + \frac{\ln(2/\epsilon)}{n-1}} + \sqrt{\frac{\ln(2/\epsilon)}{n-1}}\right)^2 \\
&= \hat{p}_{\mathrm{coll}} + \frac{2\ln(2/\epsilon)}{n-1} + 2\sqrt{\left(\hat{p}_{\mathrm{coll}}\cdot\frac{\ln(2/\epsilon)}{n-1}\right)^2 + \left(\frac{\ln(2/\epsilon)}{n-1}\right)^2} \\
&\leqslant \hat{p}_{\mathrm{coll}} + \sqrt{\frac{4\hat{p}_{\mathrm{coll}}\ln(2/\epsilon)}{n-1}} + \frac{4\ln(2/\epsilon)}{n-1}. \qquad (11)
\end{aligned}$$

where the last line follows by the elementary inequality $\sqrt{a+b}\leqslant\sqrt{a}+\sqrt{b}$. Now, from Equation (11) and Equation (7) it follows that

$$\frac{1}{n-1}\sum_{j=0}^{t}\left(n_{j+1}-n_j-1\right)p^j_{\mathrm{coll}} \leqslant \hat{p}_{\mathrm{coll}} + \sqrt{\frac{4\hat{p}_{\mathrm{coll}}\ln(2/\epsilon)}{n-1}} + \frac{4\ln(2/\epsilon)}{n-1}$$

or, equivalently, that

$$\sum_{j=0}^{t}\left(n_{j+1}-n_j\right)p^j_{\mathrm{coll}} \leqslant (n-1)\hat{p}_{\mathrm{coll}} + \sqrt{4(n-1)\hat{p}_{\mathrm{coll}}\ln(2/\epsilon)} + 4\ln(2/\epsilon) + \sum_{j=0}^{t}p^j_{\mathrm{coll}}$$

Bounding p^j_{coll} by 1 on the right-hand side and dividing both sides by n we obtain

$$\begin{aligned}
\sum_{j=0}^{t}\frac{\left(n_{j+1}-n_j\right)}{n}p^j_{\mathrm{coll}} &\leqslant \frac{n-1}{n}\hat{p}_{\mathrm{coll}} + \sqrt{\frac{4(n-1)\hat{p}_{\mathrm{coll}}\ln(2/\epsilon)}{n^2}} + \frac{4\ln(2/\epsilon)+t+1}{n} \\
&\leqslant \hat{p}_{\mathrm{coll}} + \sqrt{\frac{4\hat{p}_{\mathrm{coll}}\ln(2/\epsilon)}{n}} + \frac{4\ln(2/\epsilon)+t+1}{n} \qquad (12)
\end{aligned}$$

with probability $1-\epsilon$. To derive a bound in terms of entropies, we rewrite the right-hand side in a relative-error form

$$\sum_{j=0}^{t} \frac{(n_{j+1}-n_j)}{n} p_{\mathrm{coll}}^{j} \leqslant \hat{p}_{\mathrm{coll}} \left(1+\sqrt{\frac{4\ln(2/\epsilon)}{n\hat{p}_{\mathrm{coll}}}}+\frac{4\ln(2/\epsilon)+t+1}{n\hat{p}_{\mathrm{coll}}}\right).$$

This inequality can be rewritten, by taking the logarithm of both sides, as

$$-\log\left(\sum_{j=0}^{t} \frac{(n_{j+1}-n_j)}{n} p_{\mathrm{coll}}^{j}\right) \geqslant -\log \hat{p}_{\mathrm{coll}} - \log\left(1+\sqrt{\frac{4\ln(2/\epsilon)}{n\hat{p}_{\mathrm{coll}}}}+\frac{4\ln(2/\epsilon)+t+1}{n\hat{p}_{\mathrm{coll}}}\right).$$

The right-hand side can be bounded in a simper way by the elementary inequality $\log(1+u) \leqslant \frac{u}{\ln 2}$ valid for all $u > -1$. This gives us (with probability $1-\epsilon$)

$$-\log\left(\sum_{j=0}^{t} \frac{(n_{j+1}-n_j)}{n} p_{\mathrm{coll}}^{j}\right) \geqslant -\log \hat{p}_{\mathrm{coll}} - \sqrt{\frac{4\log(2/\epsilon)}{n\hat{p}_{\mathrm{coll}}}} - \frac{4\log(2/\epsilon)+\frac{t+1}{\ln 2}}{n\hat{p}_{\mathrm{coll}}} \tag{13}$$

Consider now the left-hand side. By applying the Jensen Inequality to the convex function $u \to -\log u$, arguments $u_j = p_{\mathrm{coll}}^{j}$ and weights $\alpha_j = \frac{n_{j+1}-n_j}{n}$ we obtain

$$\begin{aligned}\sum_{j=0}^{t} \frac{(n_{j+1}-n_j)}{n}\left(-\log\left(p_{\mathrm{coll}}^{j}\right)\right) &\geqslant -\log\left(\sum_{j=0}^{t} \frac{(n_{j+1}-n_j)}{n} p_{\mathrm{coll}}^{j}\right) \\ &\geqslant -\log \hat{p}_{\mathrm{coll}} - \sqrt{\frac{4\log(2/\epsilon)}{n\hat{p}_{\mathrm{coll}}}} - \frac{4\log(2/\epsilon)+\frac{t+1}{\ln 2}}{n\hat{p}_{\mathrm{coll}}}. \end{aligned} \tag{14}$$

Since $X_1,\ldots,X_n$ are independent, we have

$$\begin{aligned}\mathrm{CP}\left(X_1,\ldots,X_n\right) &= \prod_{i=1}^{n} \mathrm{CP}\left(()\, X_i\right) \\ &= \prod_{j=0}^{t}\prod_{i=n_j}^{n_{j+1}-1} \mathrm{CP}(X_i) \\ &= \prod_{j=0}^{t}\prod_{i=n_j}^{n_{j+1}-1} \mathrm{CP}(Y_j) \\ &= \prod_{j=0}^{t} \left(p_{\mathrm{coll}}^{j}\right)^{n_{j+1}-n_j},\end{aligned}$$

and therefore the collision entropy per bit equals

$$-\frac{\log \mathrm{CP}\left(X_1,\ldots,X_n\right)}{n} = \sum_{j=0}^{t} \frac{n_{j+1}-n_j}{n}\cdot\left(-\log p_{\mathrm{coll}}^{j}\right),$$

which combined with Equation (14) finishes the proof. To obtain the second bound on δ, we simply skip the step just after Equation (10) and proceed with the unknown parameter $\mathbf{E}\hat{p}_{\mathrm{coll}}$. ◀

4 Application to On-line Estimation

Consider a source which outputs 10-bit samples. Suppose that the entropy rate is $r = \frac{2}{10}$. Suppose we want to generate a key of length $\ell = 256$ which is at most $\epsilon = 2^{-112}$-far from the uniform distribution. If we use universal hashing then we need $\ell + 2\log(1/\epsilon)$ entropy bits, that is 480 entropy bits. This, we need at least 240 samples.

Suppose we have collected $n = 240$ samples. Taking the error into account, we cocnlude that we can generate $\ell = 120$ bits with security $\epsilon = 2^{-60}$. Thus, the quality goes down by a factor of 2, at the price of having provable security.

5 Application to Mixed Sources

Imagine a stream of data, where a few different independent sources contributes to every consecutive block. For example in [VSH11] the authors consider using an iPhone accelerometer as a source, which outputs readings from three axes X,Y and Z. The corresponding random process may be described as

$$V_1, V_2, \ldots, V_{3n} = X_1, Y_1, Z_1, X_2, Y_2, Z_2, \ldots, X_n, Y_n, Z_n$$

It can be seen that if in our collision counting estimator we compare $V_i = V_{i-3}$ instead of $V_i = V_{i-1}$ then we get a collision-entropy estimator with the same convergence bounds (up to a constant factor). Indeed, the random variables $Z_i = \mathbf{1}_{\{V_i = V_{i=3}\}}$ are all independent, and thus the estimator doesn't depend on the order. We can now imagine that the order is slightly different

$$V_{\sigma_1}, V_{\sigma_2}, \ldots, V_{\sigma_{3n}} = X_1, X_2, \ldots, X_n, Y_1, Y_2, \ldots, Y_n, Z_1, Z_2, \ldots, Z_n$$

which corresponds to $t = 2$ switches (the distribution changes two times). Therefore, out bounds apply.

6 Conclusion

We have shown that the simple collision-counting entropy estimator is (almost) as good as estimating min-entropy in terms of accuracy, but it is very efficient and robust against changing the source distribution. The assumption that consecutive outputs are independent is not that restrictive as it has been confirmed empirically and argued theoretically for many sources.

References

Ash90 Robert B. Ash, *Information theory*, Dover Publications, 1990.

BKMS09 J. Bouda, J. Krhovjak, V. Matyas, and P. Svenda, *Towards true random number generation in mobile environments*, NordSec, 2009.

BL05 Marco Bucci and Raimondo Luzzi, *Design of testable random bit generators*, CHES 2005, vol. 3659, 2005, pp. 147–156 (English).

BS Neel Bedekar and Chiranjit Shee, *A novel approach to true random number generation in wearable computing environments using mems sensors.*

BST03 B. Barak, R. Shaltiel, and E. Tromer, *True random number generators secure in a changing environment*, CHES, 2003.

Cac97 C. Cachin, *Smooth entropy and renyi entropy*, EUROCRYPT, 1997.

dRHG+99 T. de Raadt, N. Hallqvist, A. Grabowski, A. D. Keromytis, and N. Provos, *Cryptography in OpenBSD: An overview*, Proceedings of the Annual Conference on USENIX Annual Technical Conference (Berkeley, CA, USA), ATEC '99, USENIX Association, 1999, pp. 33–33.

GPR06 Zvi Gutterman, Benny Pinkas, and Tzachy Reinman, *Analysis of the linux random number generator*, Proceedings of the 2006 IEEE Symposium on Security and Privacy (Washington, DC, USA), SP '06, IEEE Computer Society, 2006, pp. 371–385.

Haa Mads Haahr, *random.org homepage*, Online; accessed 01-July-2016.

HILL99 Johan Hastad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby, *A pseudorandom generator from any one-way function*, SIAM J. Comput. **28** (1999), no. 4, 1364–1396.

Hol06 Thomas Holenstein, *Pseudorandom generators from one-way functions: A simple construction for any hardness*, TCC 2006, Lecture Notes in Computer Science, vol. 3876, 2006, pp. 443–461.

JK99 B. Jun and P Kocher, *The intel random number generator*, white paper prepared for Intel corporation, 1999.

KKHD14 D. Kaplan, S. Kedmi, R. Hay, and A. Dayan, *Attacking the linux prng on android: Weaknesses in seeding of entropic pools and low boot-time entropy*, 8th USENIX Workshop on Offensive Technologies (WOOT 14) (San Diego, CA), USENIX Association, 2014.

LPR11 Cédric Lauradoux, Julien Ponge, and Andrea Röck, *Online Entropy Estimation for Non-Binary Sources and Applications on iPhone*, Rapport de recherche, Inria, June 2011.

LRSV12 P. Lacharme, A. Röck, V. Strubel, and M. Videau, *The linux pseudorandom number generator revisited*, Cryptology ePrint Archive, Report 2012/251, 2012, http://eprint.iacr.org/.

Mau92 Ueli Maurer, *A universal statistical test for random bit generators*, Journal of cryptology **5** (1992), 89–105.

RW05 Renato Renner and Stefan Wolf, *Simple and tight bounds for information reconciliation and privacy amplification*, ASIACRYPT'05, Springer-Verlag, 2005, pp. 199–216.

Sha48 C. E. Shannon, *A mathematical theory of communication*, Bell system technical journal **27** (1948).

Sun09 Berk Sunar, *True random number generators for cryptography*, Cryptographic Engineering (ÇetinKaya Koç, ed.), Springer US, 2009, pp. 55–73 (English).

TBK+ Meltem Sönmez Turan, Elaine Barker, John Kelsey, Kerry A. McKay, Baish Mary L., and Mike Boyle.

vN51 John von Neumann, *Various Techniques Used in Connection with Random Digits*, J. Res. Nat. Bur. Stand. **12** (1951), 36–38.

VSH11 J. Voris, N. Saxena, and T. Halevi, *Accelerometers and randomness: Perfect together*, WiSec '11, ACM, 2011, pp. 115–126.

Wal John Walker, *Hotbits homepage*, Online; accessed 01-July-2016.

A Inefficiency of Plugin Estimators

Let X be an m-bit distribution. Suppose that we want to estimate $\mathbf{P}_X$ from i.i.d samples $X_1,\ldots,X_n$, and use this estimate in the entropy formula. Let $\hat{X}$ be the random variable corresponding to the empirical distribution of n samples, that is

$$\forall x: \quad \Pr[\hat{X}=x] = \frac{1}{n}\sum_{i=1}^{n}\mathbf{1}_{\{X_i=x\}}.$$

We want to use the estimate

$$\mathbf{H}_\infty(X) \approx \mathbf{H}_\infty(\hat{X}).$$

Consider the case when X is uniform. Suppose that we want the absolute error to be at most γ, that is

$$\left|\mathbf{H}_{\infty}(X) - \mathbf{H}_{\infty}(\hat{X})\right| \leqslant \gamma.$$

According to the min-entropy definition, this means that

$$\left|m + \max_{x} \log\left(\mathbf{P}_{\hat{X}}(x)\right)\right| \leqslant \gamma.$$

which is equivalent to

$$m - \gamma \leqslant \max_{x} \log\left(\mathbf{P}_{\hat{X}}(x)\right) \leqslant -m + \gamma.$$

In particular,

$$\forall x : \quad \mathbf{P}_{\hat{X}}(x) \leqslant 2^{-n+\gamma} = 2^{\gamma} \cdot \mathbf{P}_{X}(x).$$

This means that we need to estimate the probability mass function $\mathbf{P}_X(x)$ up to a relative error $\delta = 2^{\gamma} - 1$. According to the Chernoff Bound, with fixed x and n samples we get the error probability $\exp(-3\mathbf{P}_X(x)\delta^2) \leqslant \exp(-3 \cdot n2^{-m}\delta^2)$ for some c. Thus, to get the error term below ϵ, we need $\delta = O\left(\sqrt{2^m \log(1/\epsilon)/3n}\right)$. Even for a pretty weak bound $\gamma = 1$ (an error of 1 bit) we need $\delta = 1$ which means $n > 2^m \log(1/\epsilon)/3$ samples.